1. Introduction:

K nearest neighbors (KNN) and Decision Trees are two different machine learning algorithms.

KNN - A learning algorithm that classifies instances based on the majority class of their k nearest neighbors in the future space. This makes it most effective when classes are well-separated.

Decision trees - Use a hierarchical approach to split the feature space making them more interpretable and effective for capturing non-linear relationships.

The objective of the study is to compare the performance of these two algorithms across different datasets to understand how dataset complexity might impact the different models' performances.

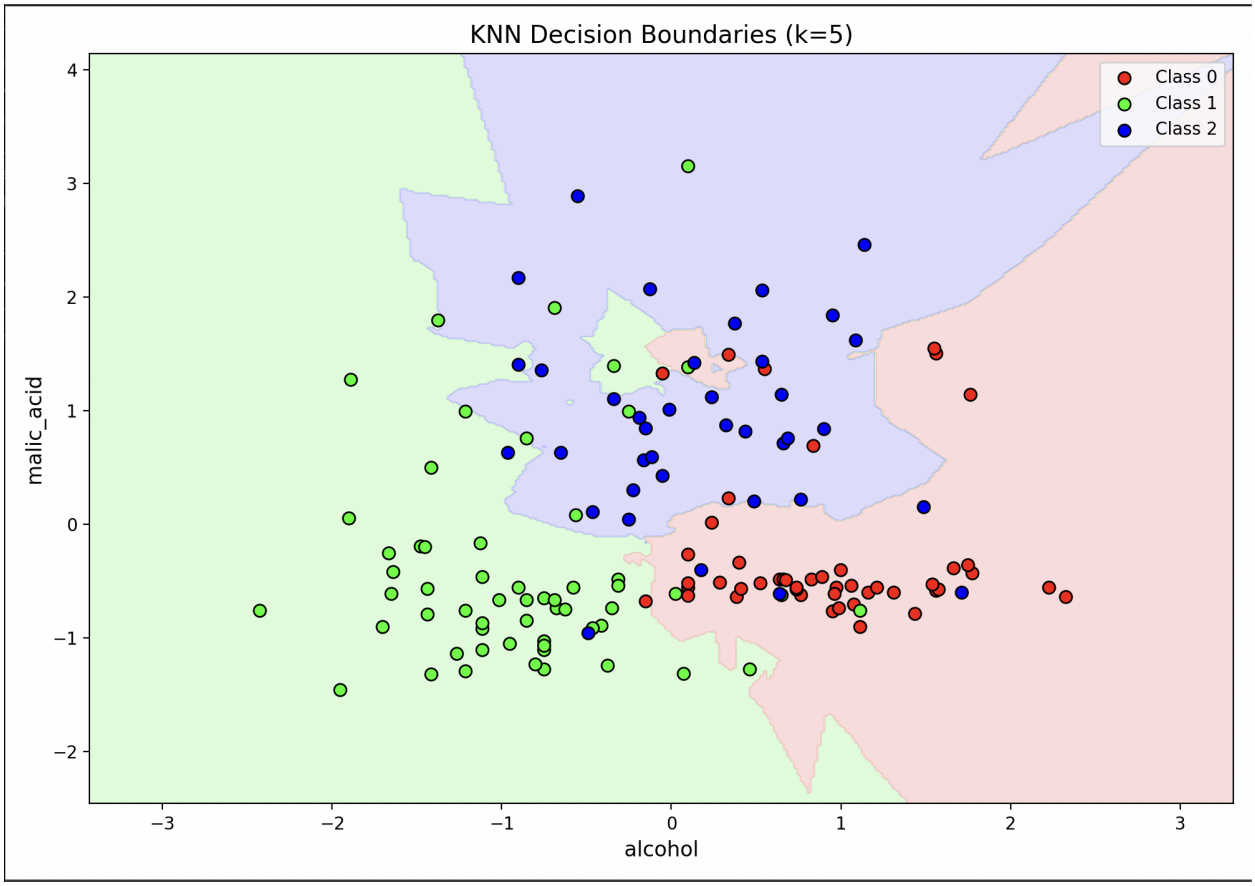
2. Method:

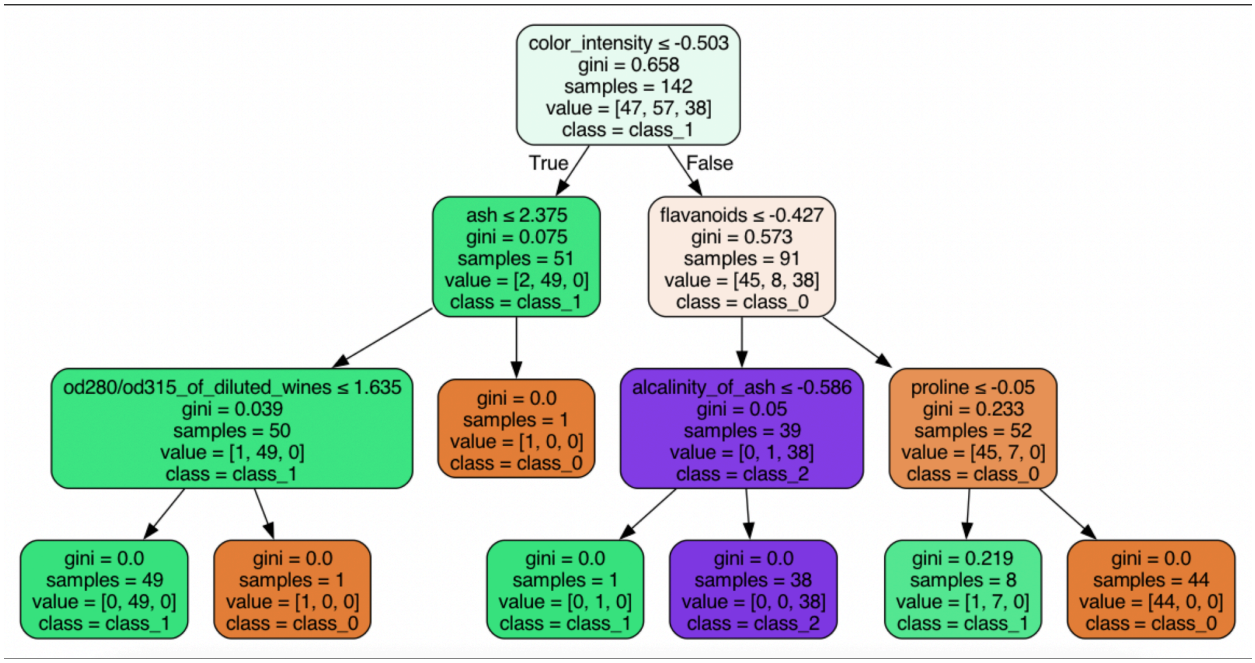
I looked at both the Wine dataset (13 features, 3 classes, 178 samples) and the Iris dataset (4 features, 3 classes, 150 samples). Both of these datasets come from scikit-learn.

Both datasets were split using an 80/20 train-test split. Features were standardized using StandardScalar to ensure a fair comparison, this is especially important for KNN's distance-based approach.

For KNN, I tested values between 1 to 11 to understand how the distance of neighbors taken into consideration affects the results. For decision trees, I tested maximum depth from 1 to 10 to analyze the complexity-performance relationship of the tree. Weighted averages were used for precision, recall and F1 scores to handle the multiclass datasets with confusion matrices.

3. + 4: Results and Discussion

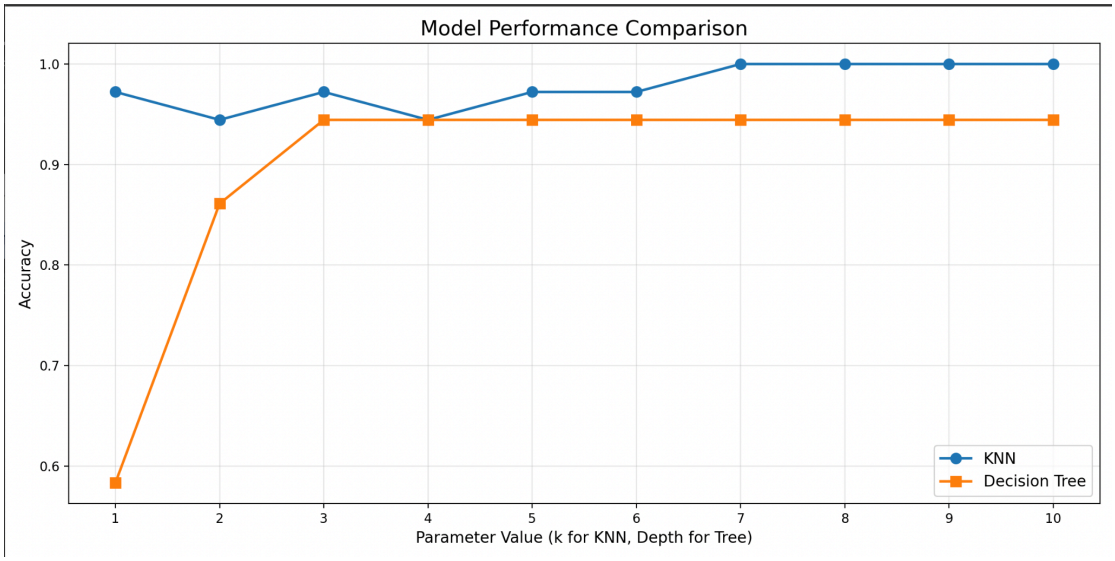

KNN Decision Boundaries (k=5)
Class 0
Class 1
Class 2
malic_acid
alcohol
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2
3


color_intensity ≤ -0.503
gini = 0.658
samples = 142
value = [47, 57, 38]
class = class_1
True
False
ash ≤ 2.375
gini = 0.075
samples = 51
value = [2, 49, 0]
class = class_1
flavanoids ≤ -0.427
gini = 0.573
samples = 91
value = [45, 8, 38]
class = class_0
od280/od315_of_diluted_wines ≤ 1.635
gini = 0.039
samples = 50
value = [1, 49, 0]
class = class_1
gini = 0.0
samples = 1
value = [1, 0, 0]
class = class_0
alcalinity_of_ash ≤ -0.586
gini = 0.05
samples = 39
value = [0, 1, 38]
class = class_2
proline ≤ -0.05
gini = 0.233
samples = 52
value = [45, 7, 0]
class = class_0
gini = 0.0
samples = 49
value = [0, 49, 0]
class = class_1
gini = 0.0
samples = 1
value = [1, 0, 0]
class = class_0
gini = 0.0
samples = 1
value = [0, 1, 0]
class = class_1
gini = 0.0
samples = 38
value = [0, 0, 38]
class = class_2
gini = 0.219
samples = 8
value = [1, 7, 0]
class = class_1
gini = 0.0
samples = 44
value = [44, 0, 0]
class = class_0

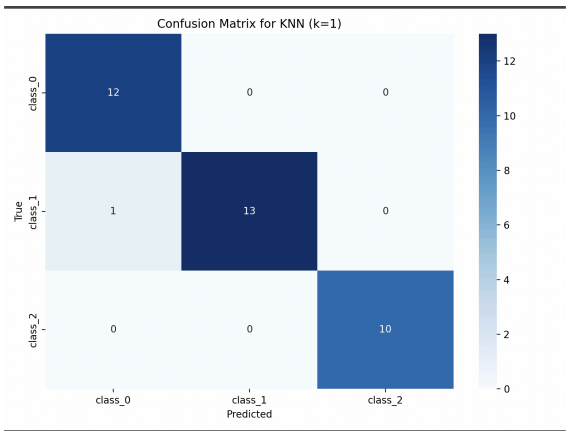
Wine dataset Results -

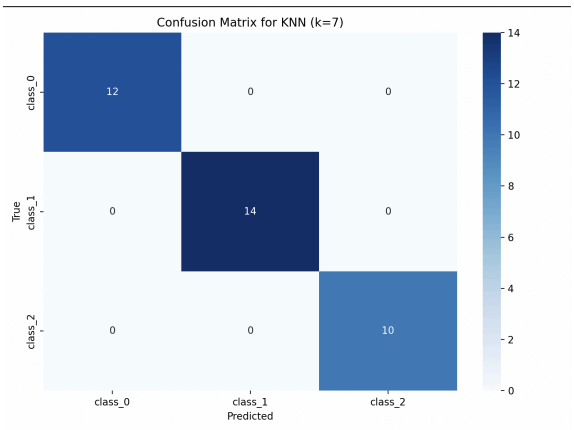
The wine dataset showed a big difference in the performance of KNN and Decision trees. KNN achieved perfect classification for k values 7-10, with the best performance coming from k at 7 as the F1 score was at 1.00. In contrast, Decision trees reached their optimal performance at depth of 3 with an F1 score at 0.9450 and showed no improvements with increasing depth (See image below)

Wine dataset performance comparison visualization:



Looking at the confusion matrices for KNN at 1 vs 7:





Showing that K at 1 had a misclassification whereas 7 (and above) were perfect.

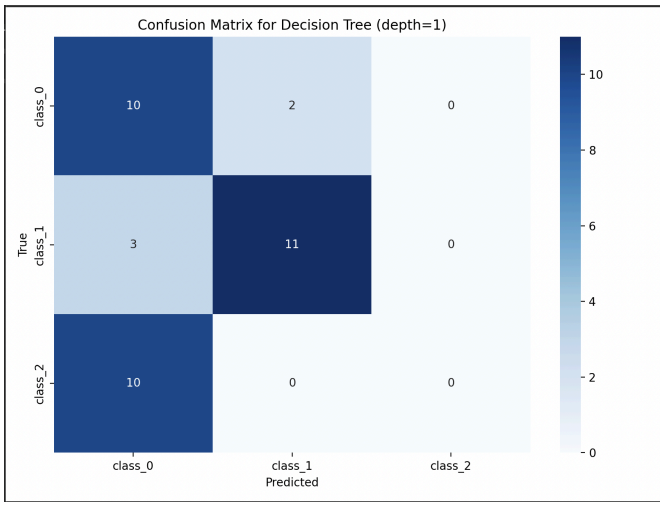
--- KNN with k=1 ---

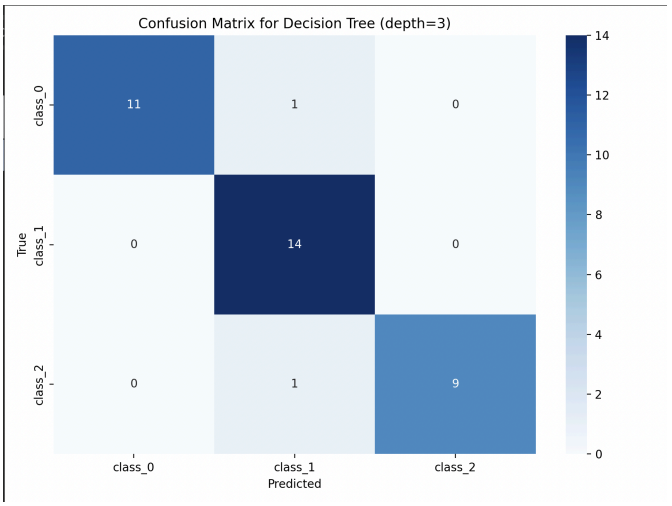
KNN (k=1) Performance:
  Accuracy: 0.9722
  Precision: 0.9744
  Recall: 0.9722
  F1 Score: 0.9723

--- KNN with k=7 ---

KNN (k=7) Performance:
  Accuracy: 1.0000
  Precision: 1.0000
  Recall: 1.0000
  F1 Score: 1.0000

Confusion matrix for decision tree depth 1 vs 3:





Looking at decision tree with depth at 1, there are many more mistakes in the model vs at its optimal level of 3:

Decision Tree (depth=1) Performance:
  Accuracy: 0.5833
  Precision: 0.4740
  Recall: 0.5833
  F1 Score: 0.507

--- Decision Tree with depth=3 ---

Decision Tree (depth=3) Performance:
  Accuracy: 0.9444
  Precision: 0.9514
  Recall: 0.9444
  F1 Score: 0.9450

Now, looking at the Iris dataset:

The best performance with KNN comes at k at 1,

KNN (k=1) Performance:
  Accuracy: 0.9667
  Precision: 0.9697
  Recall: 0.9667
  F1 Score: 0.9666

With a worsening performance as k increases:
KNN (k=2) Performance:
  Accuracy: 0.9333
  Precision: 0.9444
  Recall: 0.9333
  F1 Score: 0.9327

For decision trees, the optimal performance comes at depth 3:

Decision Tree (depth=1) Performance:
  Accuracy: 0.6667
  Precision: 0.5000
  Recall: 0.6667
  F1 Score: 0.5556

Decision Tree (depth=3) Performance:
  Accuracy: 0.9667
  Precision: 0.9697
  Recall: 0.9667
  F1 Score: 0.9666

Parameter sensitivity analysis:

KNN performance pattern:

- Wine: Clear performance jump as k increases maintaining perfect performance from 7 onwards.
- Iris: Multiple optimal k values, suggesting robustness of parameter choice.

Decision tree performance:

- Both datasets: Optimal performance at depth 3 for both datasets, with performance dropping at higher depths.
- Wine: More dramatic improvement from depth 1 to depth 3
- Iris: Slightly more gradual improvement as depth increased from 1 to 3.

Dataset comparison:

Dataset	**Features**	**Samples**	**Best KNN**	**Best Tree**	**Winner**
Wine	13	178	100.0%	94.4%	KNN
Iris	4	150	96.7%	96.7%	Tie

Impact of characteristics:

With wine having more dimensions (features), this seems to favor KNN's distance based approach allowing it to receive perfect classification. This means that wine classes are very well separated in high dimensions.

In contrast, Iris' data set 4 dimensional space allows both algos to perform equally, indicating that simpler features allow Decision trees to compete with KNN. Lower dimensions makes it easier for Decision trees to find effective splits to capture patterns.

5. Conclusion:

This comparative analysis reveals that KNN significantly outperforms Decision Trees on the high-dimensional Wine dataset (100% vs 94.4% accuracy), while both algorithms perform equally well on the lower-dimensional Iris dataset (96.7% accuracy each). The results demonstrate that dataset characteristics, particularly dimensionality and class separation, play a crucial role in determining algorithm performance. KNN excels when classes are well-separated

in high-dimensional spaces, while Decision Trees are more competitive on simpler, lower-dimensional datasets.

The study highlights the importance of parameter tuning and dataset-specific model selection, with KNN requiring careful k value selection and Decision Trees showing consistent optimal performance at depth=3. These findings suggest that you should consider dataset characteristics when choosing between these algorithms, with KNN being preferred for complex, high-dimensional problems and Decision Trees being suitable for simpler, interpretable solutions.